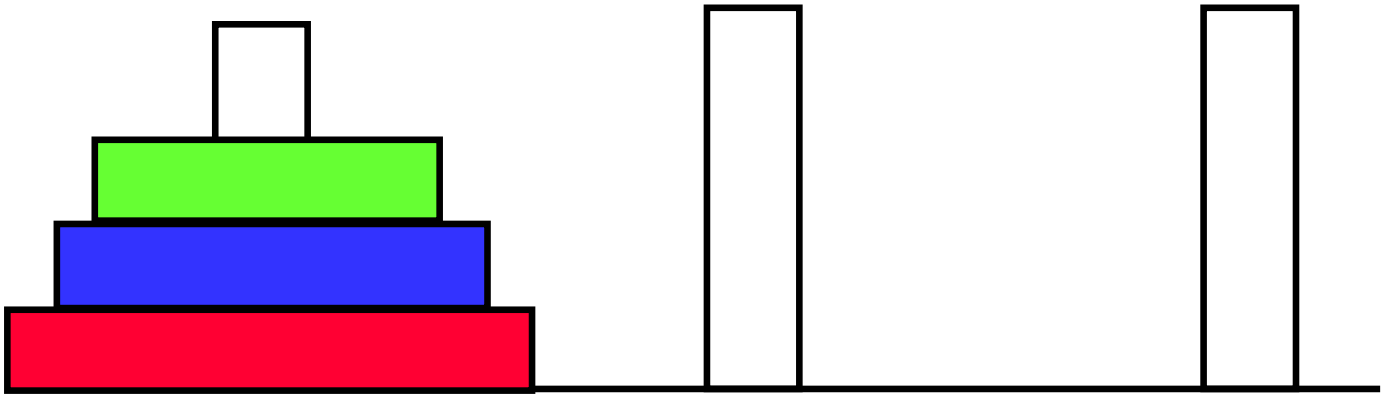
Application 4 of Stack: Tower of Hanoi

The problem is as follows:

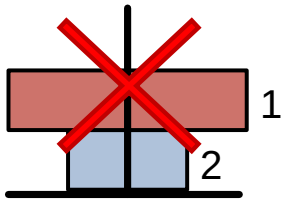
- *N* discs of decreasing size stacked on one needle & two empty needles are given.
- It is required to arrange all the discs onto a second needle in decreasing order of size.
- The Third needle may be used as temporary storage.

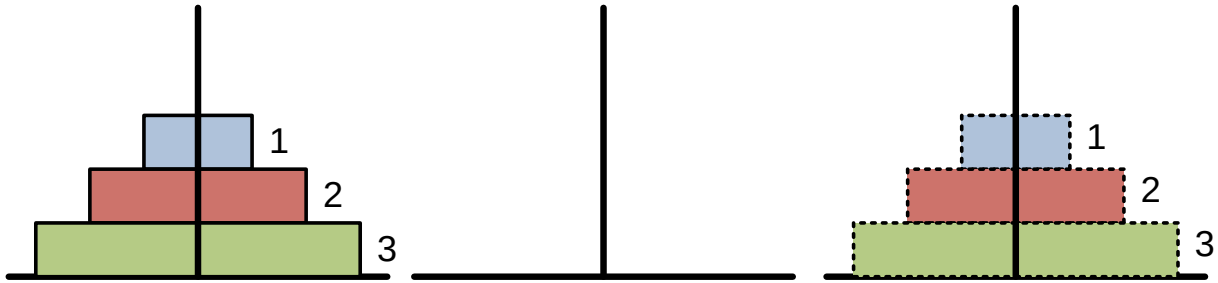


Application 4 of Stack: Tower of Hanoi (Continue)

The movement of the discs is restricted by the following rules:

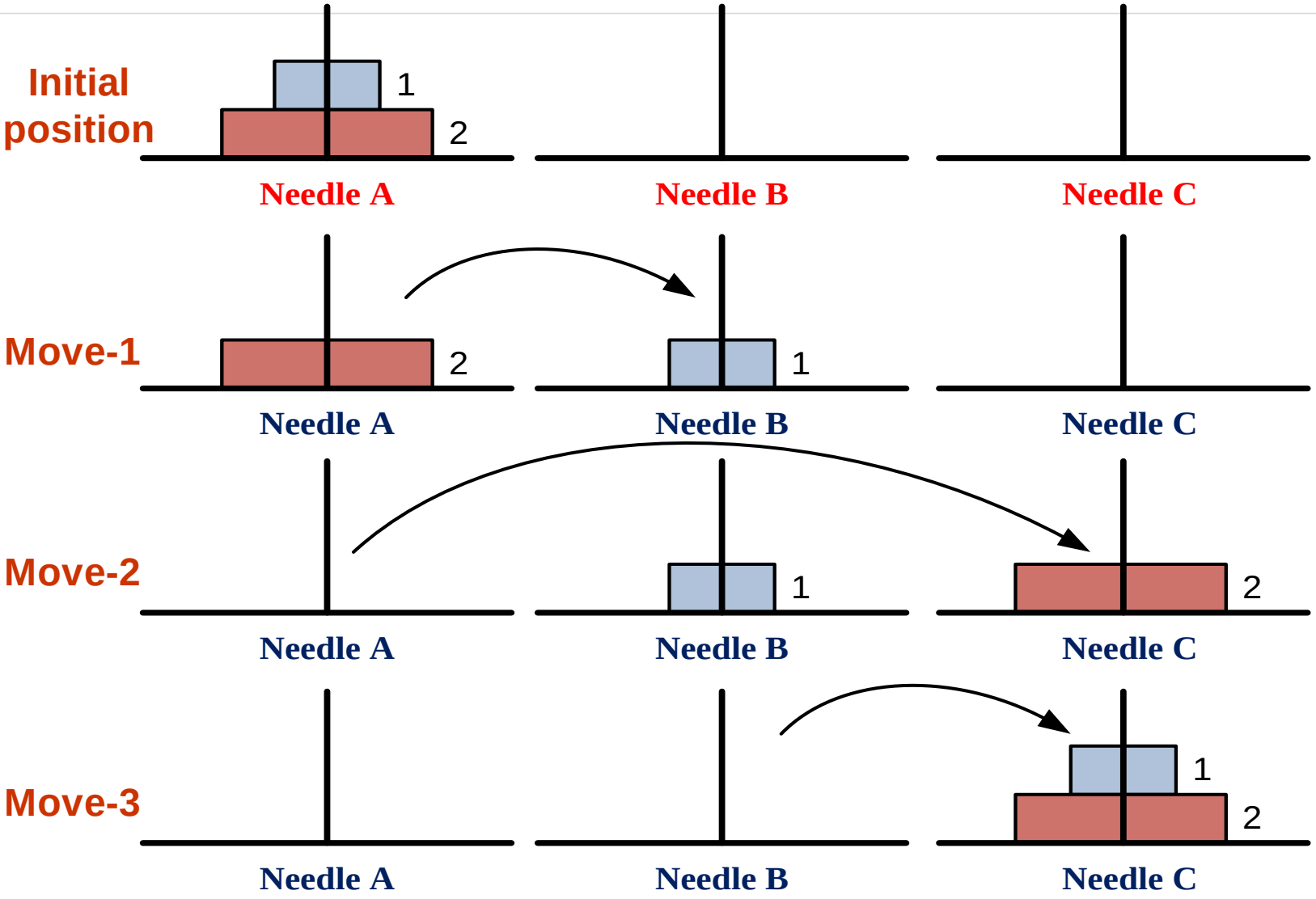
1. Only one disc may be moved at a time.
2. A disc may be moved from any needle to any other.
3. At no time may a larger disc rest upon a smaller disc.



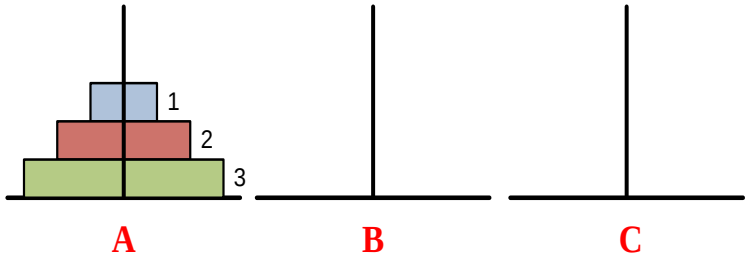


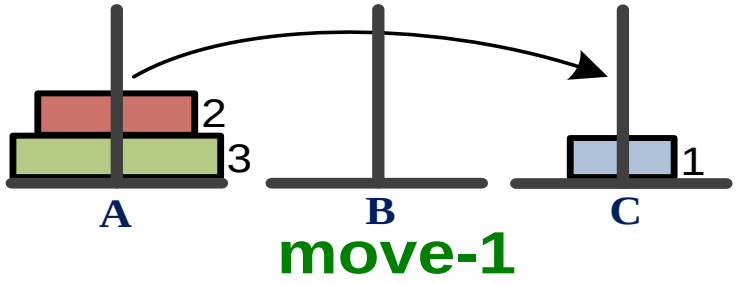
Needle A (start of problem) **Needle B (intermediate)** **Needle C (completion of problem)**

Solution of the problem (N=2)

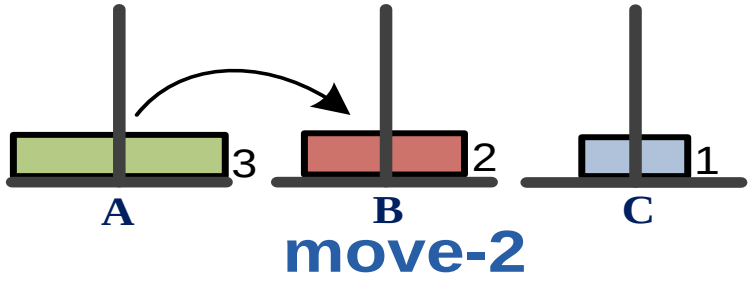


N = 3

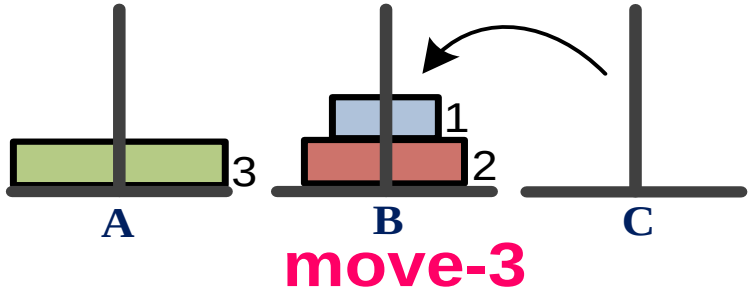




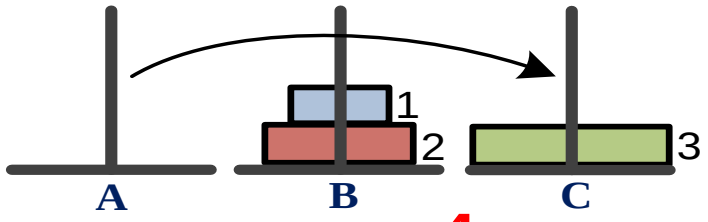
move-1



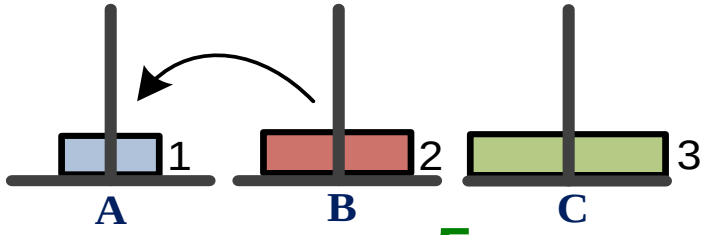
move-2



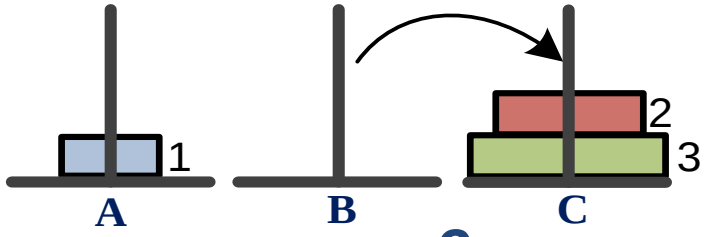
move-3



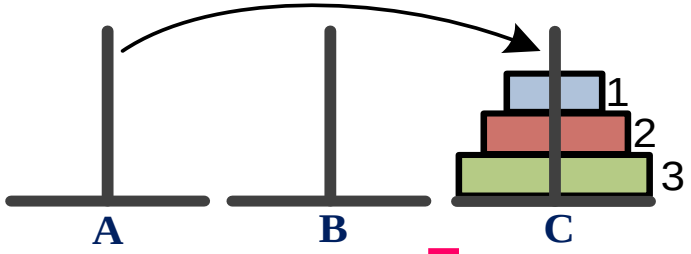
move-4



move-5



move-6



move-7

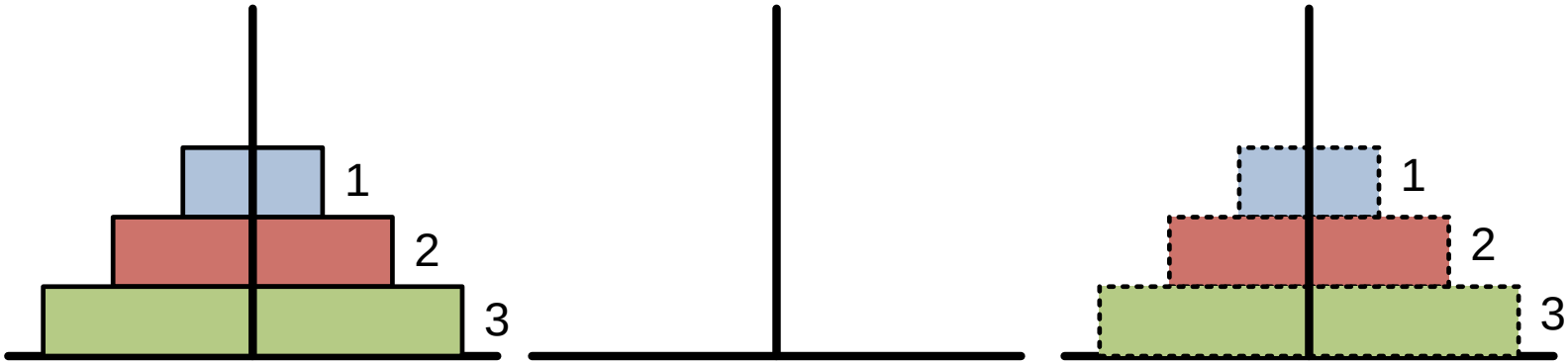
Solution

- **The solution of this problem is:**
- To move **one** disc, move it from A➞C.
- To move **two** discs, move the first disc A➞B, move the second from A➞C, then move the disc from B➞C.
- In general, the solution of the problem moving ***N*** discs from A to C has three steps:

1. Move ***N* – 1** discs from A➞B. (**source➞interm**)
2. Move ***N*th** disc from A➞C. (**source➞destination**)
3. Move ***N* – 1** discs from B➞C. (**interm➞dest**)

Problem of Tower of Hanoi

- **This problem uses STACK as a data structure for the solution. Because the problem is solved using recursion.**



Algorithm (recursive) Book

S I D

- **ToH**(source, intermediate, destination, no.of disc)

S I D

- **ToH** (**A, B, C, N**)

  if (N ≠ 0)

  {

  S I D

  ToH (**A, C, B, N-1**);

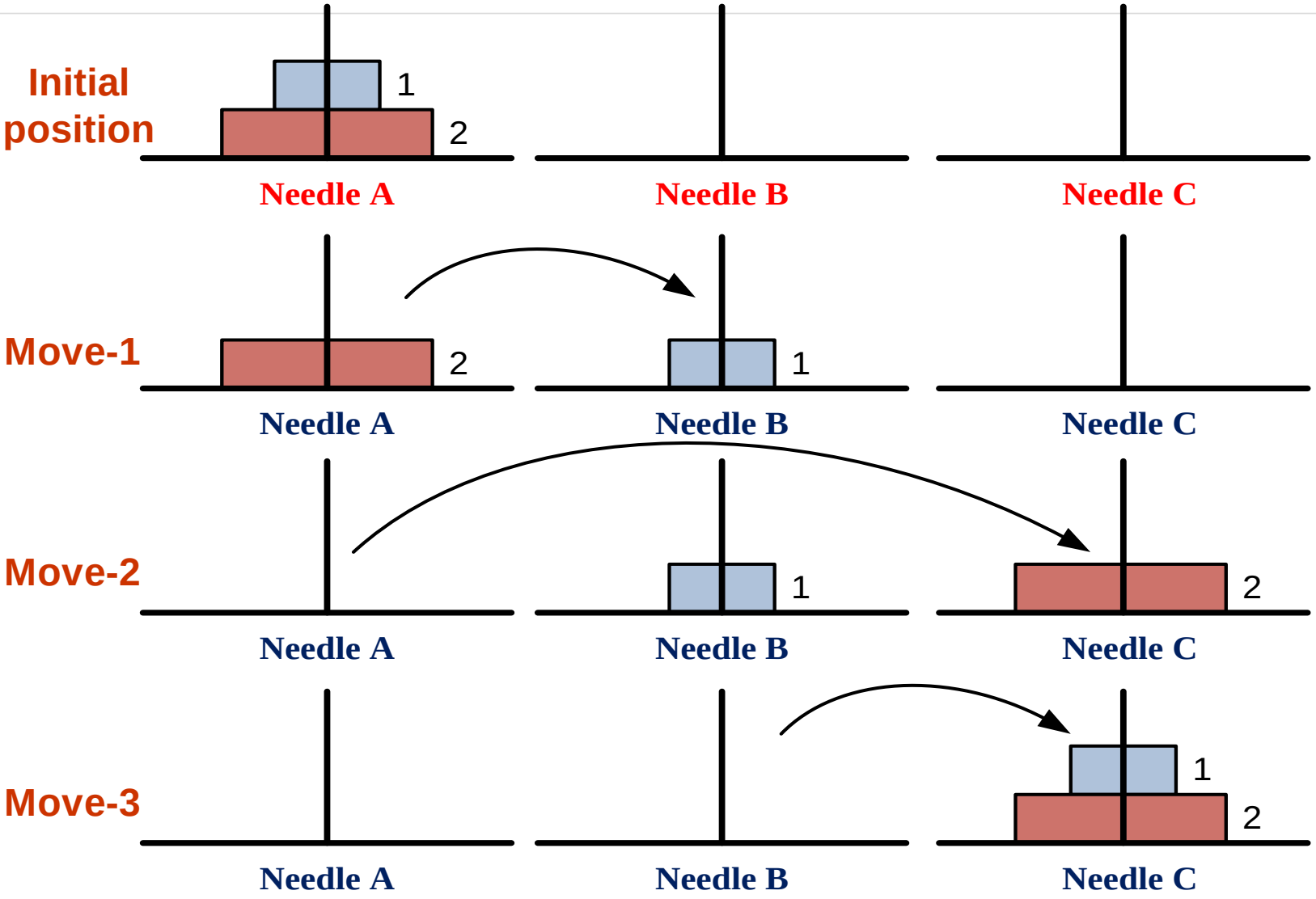
  Write (Move disc **N** from source ➞ destination);

  S I D

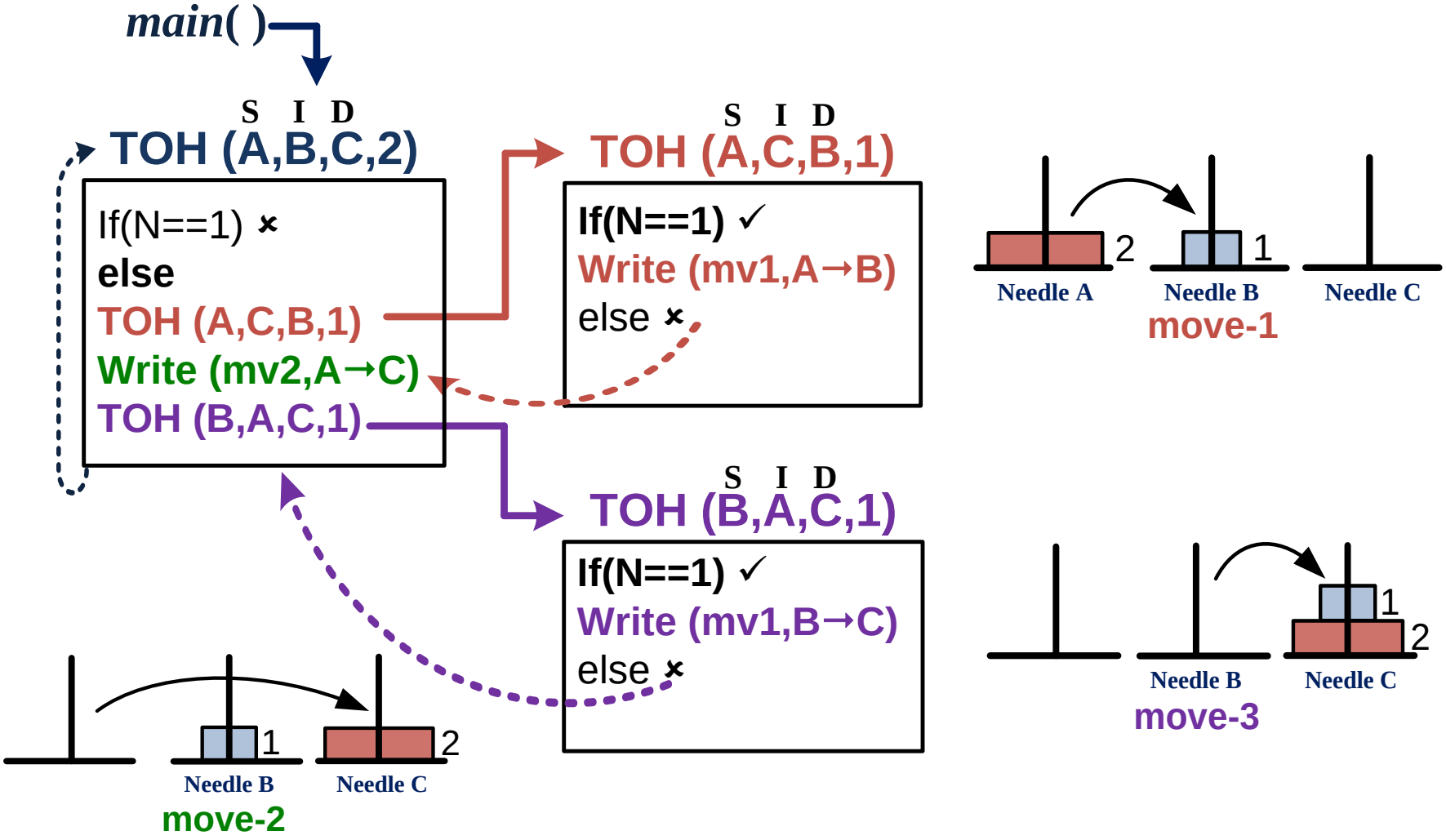
  ToH (**B, A, C, N-1**);

  }

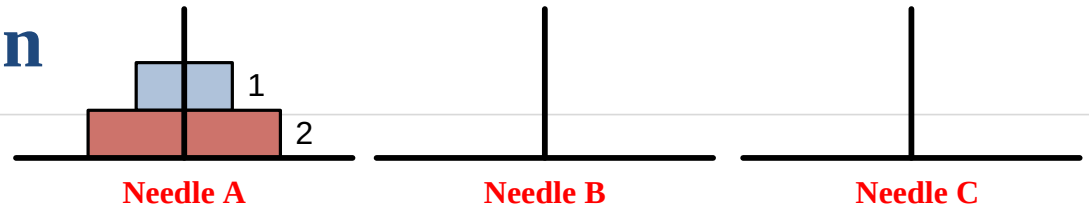
Solution of the problem (N=2)



Graphical solution of the problem (*N*=2)



N = 2 Initial Situation



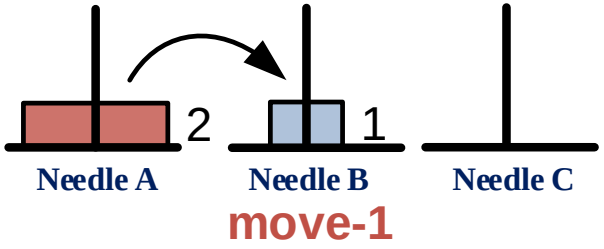
main( )

S I D
TOH (A,B,C,2)

If(N==1) ×
else
TOH (A,C,B,1)
Write (mv2,A→C)
TOH (B,A,C,1)

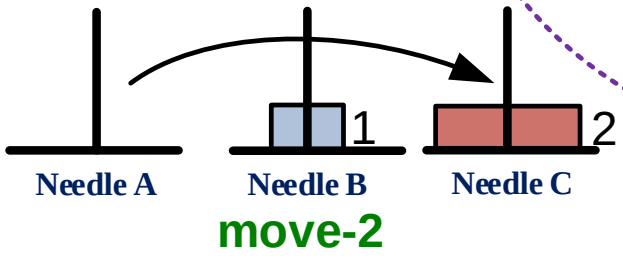
S I D
TOH (A,C,B,1)

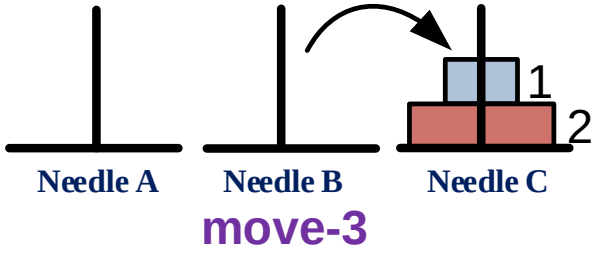
If(N==1) ✓
Write (mv1,A→B)
else ×



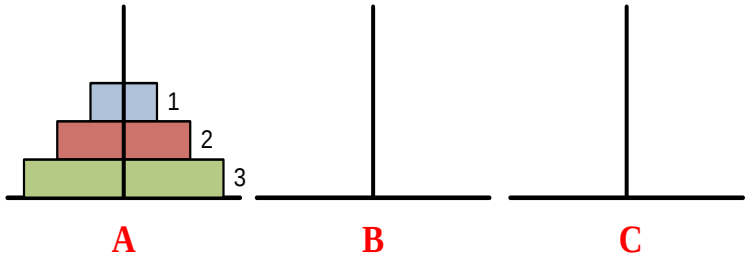
S I D
TOH (B,A,C,1)

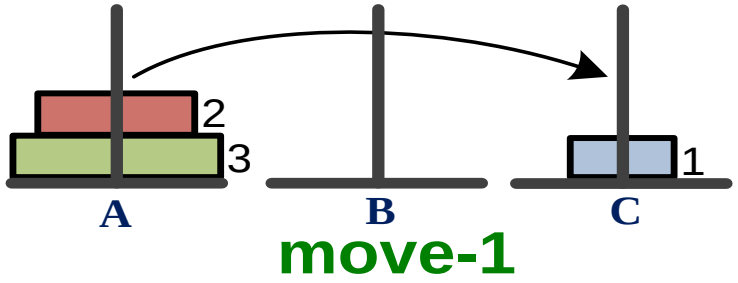
If(N==1) ✓
Write (mv1,B→C)
else ×



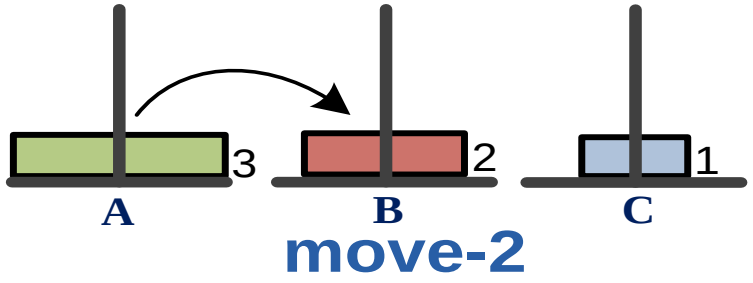


N = 3

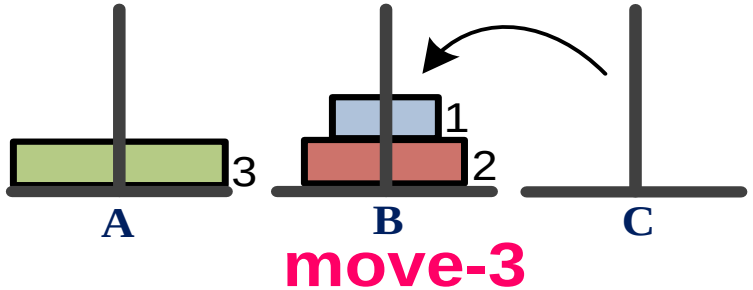




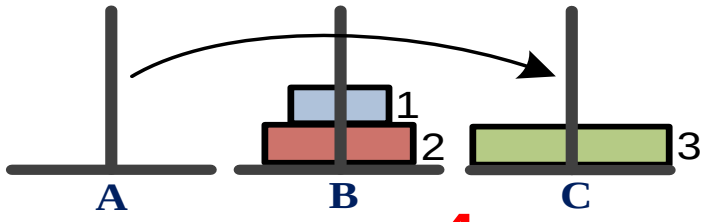
move-1



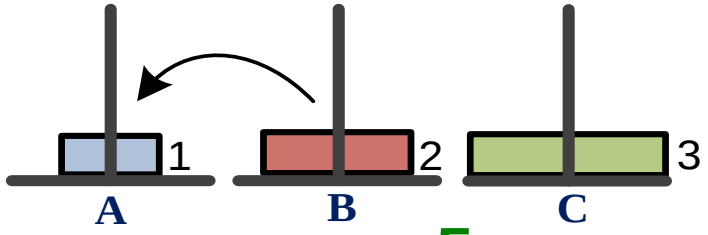
move-2



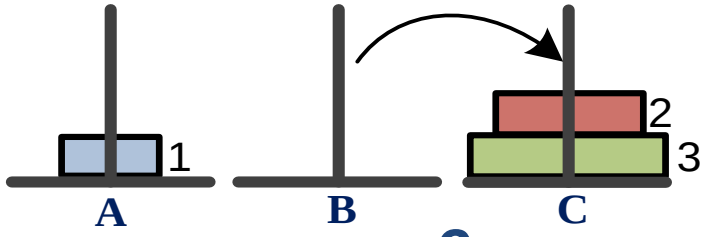
move-3



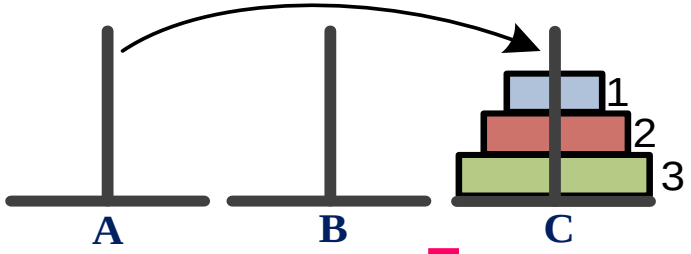
move-4



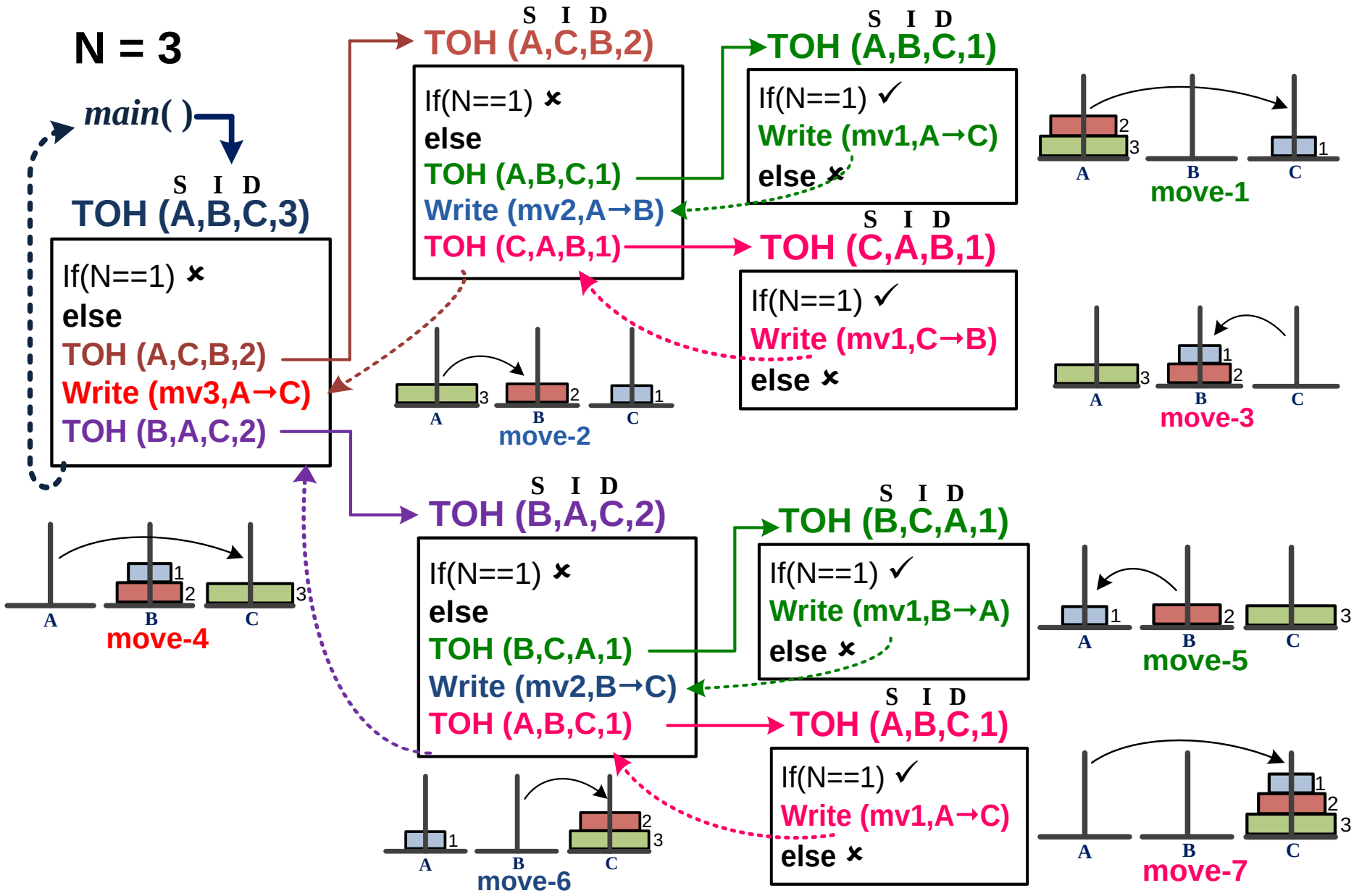
move-5



move-6



move-7


N = 3
main( )
S I D
TOH (A,B,C,3)
If(N==1) ✘
else
TOH (A,C,B,2)
Write (mv3,A→C)
TOH (B,A,C,2)
S I D
TOH (A,C,B,2)
If(N==1) ✘
else
TOH (A,B,C,1)
Write (mv2,A→B)
TOH (C,A,B,1)
S I D
TOH (A,B,C,1)
If(N==1) ✓
Write (mv1,A→C)
else ✘
move-1
S I D
TOH (C,A,B,1)
If(N==1) ✓
Write (mv1,C→B)
else ✘
move-2
move-3
S I D
TOH (B,A,C,2)
If(N==1) ✘
else
TOH (B,C,A,1)
Write (mv2,B→C)
TOH (A,B,C,1)
S I D
TOH (B,C,A,1)
If(N==1) ✓
Write (mv1,B→A)
else ✘
move-4
move-5
S I D
TOH (A,B,C,1)
If(N==1) ✓
Write (mv1,A→C)
else ✘
move-6
move-7
A
B
C

N = 3

main( )

S I D
TOH (A,B,C,3)

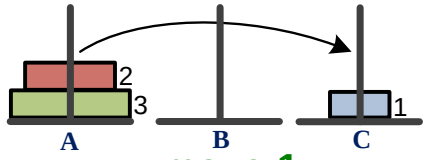
If(N==1) ✗
else
TOH (A,C,B,2)
Write (mv3,A→C)
TOH (B,A,C,2)

S I D
TOH (A,C,B,2)

If(N==1) ✗
else
TOH (A,B,C,1)
Write (mv2,A→B)
TOH (C,A,B,1)

S I D
TOH (A,B,C,1)

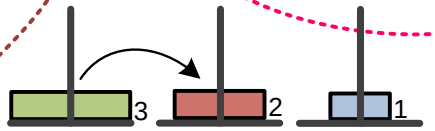
If(N==1) ✓
Write (mv1,A→C)
else ✗



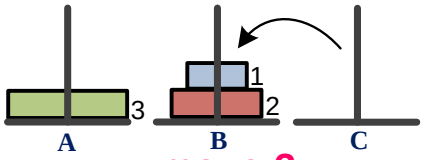
move-1

S I D
TOH (C,A,B,1)

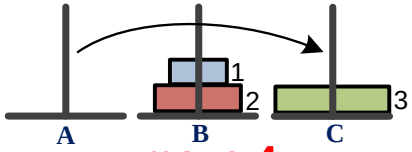
If(N==1) ✓
Write (mv1,C→B)
else ✗



move-2



move-3



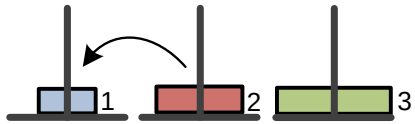
move-4

S I D
TOH (B,A,C,2)

If(N==1) ✗
else
TOH (B,C,A,1)
Write (mv2,B→C)
TOH (A,B,C,1)

S I D
TOH (B,C,A,1)

If(N==1) ✓
Write (mv1,B→A)
else ✗

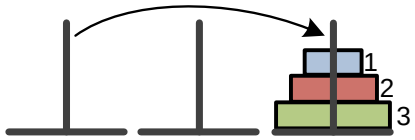


move-5

S I D
TOH (A,B,C,1)

If(N==1) ✓
Write (mv1,A→C)
else ✗

move-6



move-7